Serial No. 10/032,567   2
Docket No. YOR920010366US2

AMENDMENT TO THE CLAIMS:

1.   (Currently amended) A method for statically detecting a datarace condition in a
multithreaded application, said method comprising:

      inputting a set of input information;

      processing the set of input information by comparing threads that may execute statements
in a statement pair; and

      outputting a statement conflict set that identifies the statement pairs having ~~whose~~
execution instances which definitely or potentially cause dataraces, without executing the
multithreaded application.

2.   (Original) The method of claim 1, wherein the processing comprises:

      selectively evaluating the input information with an IsPotentialDR relation; and

      selectively evaluating the input information with an IsDefiniteDR relation.

3.   (Original) The method of claim 2, wherein, for a given pair of reference expressions, the
IsPotentialDR relation comprises:

      determining whether the reference expressions might be executed by different threads
(negation of DefSameThreadObj);

      determining whether the reference expressions might access the same field of the same
object; and

      determining whether the reference expressions might not be mutually synchronized
(negation of DefSync).

4.   (Previously presented) The method of claim 2, wherein, for a given pair of reference
expressions, the IsDefiniteDR relation comprises:

      determining whether the reference expressions cannot be executed by the same thread
(negation of PossSameThreadObj);

Serial No. 10/032,567   3
Docket No. YOR920010366US2

     determining whether the reference expressions must access the same field of the same object;

     determining whether the reference expressions cannot be mutually synchronized (negation of PossSync); and

     determining whether the reference expressions must execute.

5.   (Original) The method of claim 1, wherein the set of input information comprises a multithreaded context graph (multithreaded context graphs).

6.   (Previously presented) The method of claim 5, wherein the multithreaded context graph comprises an interprocedural call graph having each of a plurality of synchronized blocks as a separate node.

7.   (Previously presented) The method of claim 5, wherein the multithreaded context graph comprises an interprocedural call graph having each of a plurality of synchronized methods as a separate node.

8.   (Previously presented) The method of claim 1, further comprising:
     performing dynamic datarace detection on the Statement Conflict Set.

9.   (Previously presented) The method of claim 1, further comprising:
     performing escape analysis to identify statements that can access memory locations accessible by more than one thread.

10.   (Original) The method of claim 1, wherein the processing comprises:
     computing a node conflict set; and
     computing the Statement Conflict Set by determining pairs of conflicting statements in the node conflict set.

Serial No. 10/032,567   4
Docket No. YOR920010366US2

11.   (Previously presented) The method of claim 10, wherein said computing the node conflict set comprises:

initializing a synchronization object set for each of a plurality of multithreaded context graphs nodes.

12.   (Previously presented) The method of claim 11, wherein said computing the node conflict set further comprises:

identifying all reachable conflicting node pairs for each thread-root node.

13.   (Previously presented) The method of claim 12, wherein said computing the node conflict set further comprises:

identifying all reachable conflicting node pairs for each distinct pair of thread-root nodes in the multithreaded context graphs; and

identifying all reachable conflicting node pairs for each thread-root node in the multithreaded context graphs that is invokeable by more than one thread.

14.   (Original) The method of claim 1, wherein the input comprises meta-information relating to a multithreaded application written in an object-oriented programming language.

15.   (Previously presented) The method of claim 1, wherein the input comprises a multithreaded context graph (MCG) for a multithreaded application written in an object-oriented programming language.

16.   (Original) The method of claim 15, wherein the input further comprises a plurality of bytecodes that collectively comprise the application.

17.   (Previously presented) A computer processing system for statically detecting a datarace

Serial No. 10/032,567   5
Docket No. YOR920010366US2

condition in a multithreaded application, comprising:

    an input interface;

    an output interface;

    a storage medium comprising the application and meta-information relating to the application; and

    a processor configured to receive the application and the meta-information, process the application and the meta-information without executing the application, and determine a statement conflict set (SCS) for the application,

    wherein said processor compares threads that may execute statements in a statement pair.

18.   (Original) The computer processing system of claim 17, wherein the meta-information comprises a multi-threaded context graph.

19.   (Original) The computer processing system of claim 17, wherein the processor is further configured to perform dynamic datarace detection on the SCS.

20.   (Previously presented) A computer program product, comprising a computer readable medium having computer code embodied therein for statically detecting a datarace condition in a multithreaded application, said computer program product comprising:

    computer readable program code devices configured to receive the application and the meta-information;

    computer readable program code devices configured to process the application and the meta-information without executing the application; and

    computer readable program code devices configured to determine a statement conflict set (SCS) for the application by comparing threads that may execute statements in a statement pair.

21.   (Previously presented) The method of claim 1, wherein said comparing said threads comprises:

Serial No. 10/032,567   6
Docket No. YOR920010366US2

tagging a statement with a set of threads that may execute said statement, and comparing

sets of threads for said statements.

22.   (Previously presented) The method of claim 1, wherein said comparing said threads

comprises:

comparing sets of locks held by threads that may execute said statements.

23.   (Currently amended) A method for statically detecting a datarace condition in a

multithreaded application, said method comprising:

inputting a set of input information;

processing the set of input information by comparing threads that may execute statements

in a statement pair;

outputting a statement conflict set that identifies the statement pairs having ~~whose~~

execution instances <u>which</u> definitely or potentially cause dataraces, without executing the

multithreaded application;

performing dynamic datarace detection on the Statement Conflict Set; and

performing escape analysis to identify statements that can access memory locations

accessible by more than one thread,

wherein said processing comprises:

computing a node conflict set; and

computing the Statement Conflict Set by determining pairs of conflicting

statements in the node conflict set,

wherein said computing the node conflict set comprises:

initializing a synchronization object set for each of a plurality of multithreaded

context graph nodes;

identifying all reachable conflicting node pairs for each thread-root node;

identifying all reachable conflicting node pairs for each distinct pair of thread-root

nodes in the multithreaded context graphs; and

Serial No. 10/032,567   7
Docket No. YOR920010366US2

      identifying all reachable conflicting node pairs for each thread-root node in the multithreaded context graphs that is invokeable by more than one thread, and

      wherein the input comprises a multithreaded context graph (MCG) for a multithreaded application written in an object-oriented programming language.